Class 7: Machine Learning 1

Sarah Mirsaidi Madjdabadi, A16890186
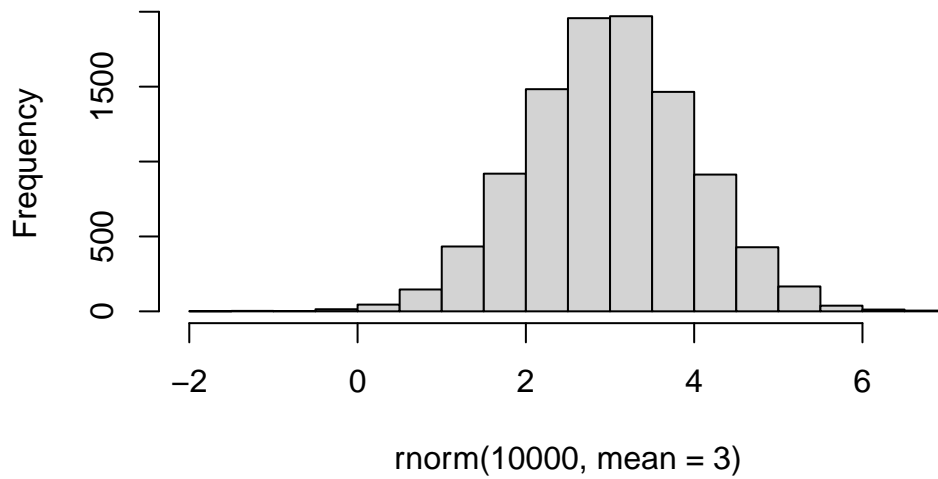
Table of contents

Today we will explore unsupervised machine learning methods starting with clustering and dimensionality reduction.

Clustering

To start let's make up some data to cluster where we know what the answer should be. The `rnorm()` function will help us here.

```
hist( rnorm(10000, mean=3) )
```

Histogram of rnorm(10000, mean = 3)

Return 30 numbers centered on -3.

```r
tmp <- c( rnorm(30, mean=-3),
  rnorm (30, mean=3))

x <- cbind(x=tmp, y=rev(tmp))

x
```

```
               x         y
 [1,] -2.085612  2.711834
 [2,] -4.379787  5.242015
 [3,] -2.807624  1.633409
 [4,] -3.457270  3.624575
 [5,] -4.327056  1.273065
 [6,] -2.489034  3.291827
 [7,] -3.573461  4.128115
 [8,] -4.842777  3.669958
 [9,] -3.329802  1.963688
[10,] -3.219435  3.597973
[11,] -5.660960  3.896151
[12,] -3.903236  3.790950
[13,] -1.868958  3.251629
```

```
[14,]  -2.333246   3.169867
[15,]  -3.544375   3.014904
[16,]  -2.097094   3.203343
[17,]  -3.333437   3.141068
[18,]  -5.859847   2.972971
[19,]  -1.794718   1.928724
[20,]  -3.829180   2.757721
[21,]  -2.620412   1.029284
[22,]  -4.141353   1.663299
[23,]  -3.382954   3.291054
[24,]  -2.387492   2.783061
[25,]  -2.863049   2.210813
[26,]  -4.175254   3.167034
[27,]  -4.153858   3.471593
[28,]  -1.262651   2.755450
[29,]  -1.957095   2.860904
[30,]  -3.491598   3.058649
[31,]   3.058649  -3.491598
[32,]   2.860904  -1.957095
[33,]   2.755450  -1.262651
[34,]   3.471593  -4.153858
[35,]   3.167034  -4.175254
[36,]   2.210813  -2.863049
[37,]   2.783061  -2.387492
[38,]   3.291054  -3.382954
[39,]   1.663299  -4.141353
[40,]   1.029284  -2.620412
[41,]   2.757721  -3.829180
[42,]   1.928724  -1.794718
[43,]   2.972971  -5.859847
[44,]   3.141068  -3.333437
[45,]   3.203343  -2.097094
[46,]   3.014904  -3.544375
[47,]   3.169867  -2.333246
[48,]   3.251629  -1.868958
[49,]   3.790950  -3.903236
[50,]   3.896151  -5.660960
[51,]   3.597973  -3.219435
[52,]   1.963688  -3.329802
[53,]   3.669958  -4.842777
[54,]   4.128115  -3.573461
[55,]   3.291827  -2.489034
[56,]   1.273065  -4.327056
```

```
[57,]  3.624575 -3.457270
[58,]  1.633409 -2.807624
[59,]  5.242015 -4.379787
[60,]  2.711834 -2.085612
```

Make a plot of x

```
plot(x)
```
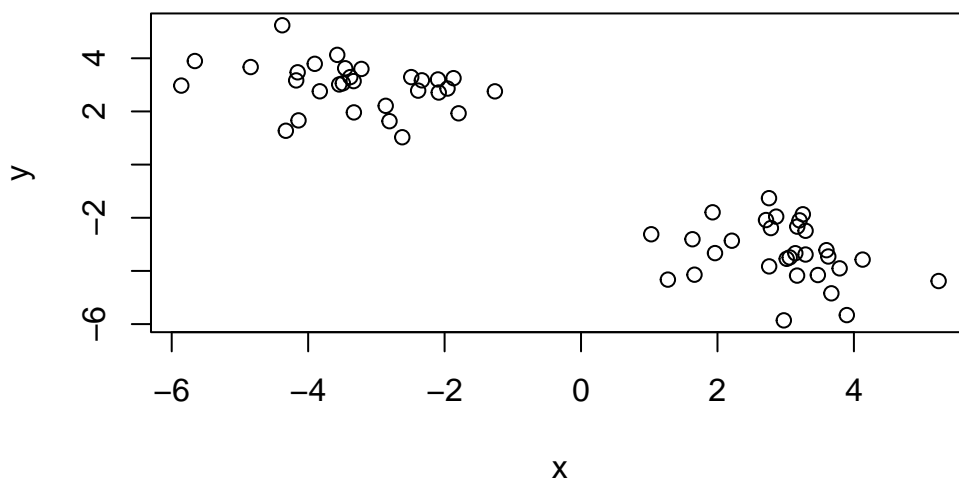

K-means

The main function in "base" R for K-means clustering is called kmeans():

```
km <- kmeans(x, centers = 2)
km
```

```
K-means clustering with 2 clusters of sizes 30, 30

Cluster means:
          x         y
1 -3.305754  2.951831
```

```
2   2.951831 -3.305754
```

```
Clustering vector:
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
Within cluster sum of squares by cluster:
[1] 59.36054 59.36054
 (between_SS / total_SS =  90.8 %)
```

```
Available components:

[1] "cluster"      "centers"      "totss"        "withinss"      "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

The `kmeans()` returns a "list" with 9 components. You can see the anmed components of any list with the `attributes()` function.

```
attributes(km)
```

```
$names
[1] "cluster"      "centers"      "totss"        "withinss"      "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```
$class
[1] "kmeans"
```

Q. How many points are in each cluster?

```
km$size
```

```
[1] 30 30
```

Q. Cluster assignment/membership vector?

```
km$cluster
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

Q. Cluster centers?

```
km$centers
```

```
          x          y
1 -3.305754   2.951831
2  2.951831  -3.305754
```

Q. Make a plot of our `kmeans()` results showing cluster alignment using different colors for each cluster/group of points and cluster centers in blue.

```r
plot(x,col=km$cluster)
points(km$centers, col="blue", pch=15, cex=2)
```
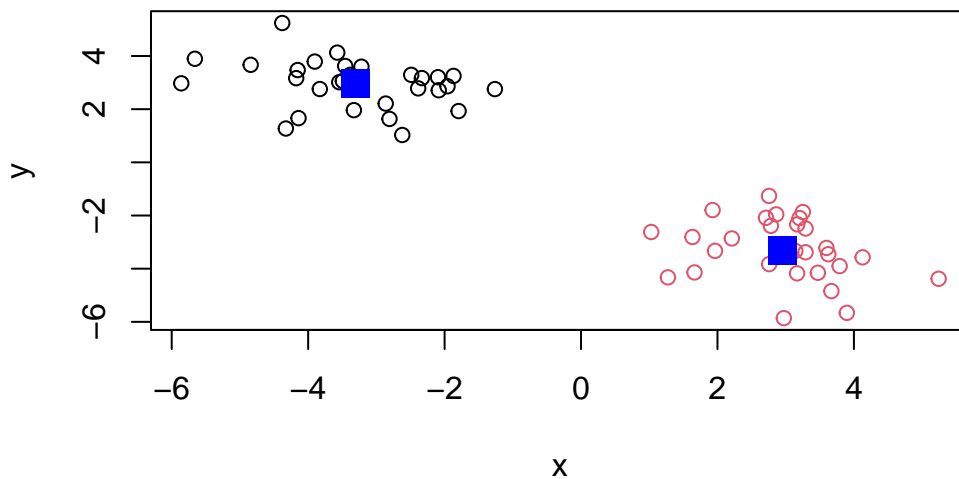

Q. Run `kmeans()` again on `x` and this cluster into 4 groups/clusters and plot the same result figure as above.

```r
km4 <- kmeans(x, centers = 4)
plot(x,col=km4$cluster)
points(km4$centers, col="blue", pch=15, cex=2)
```
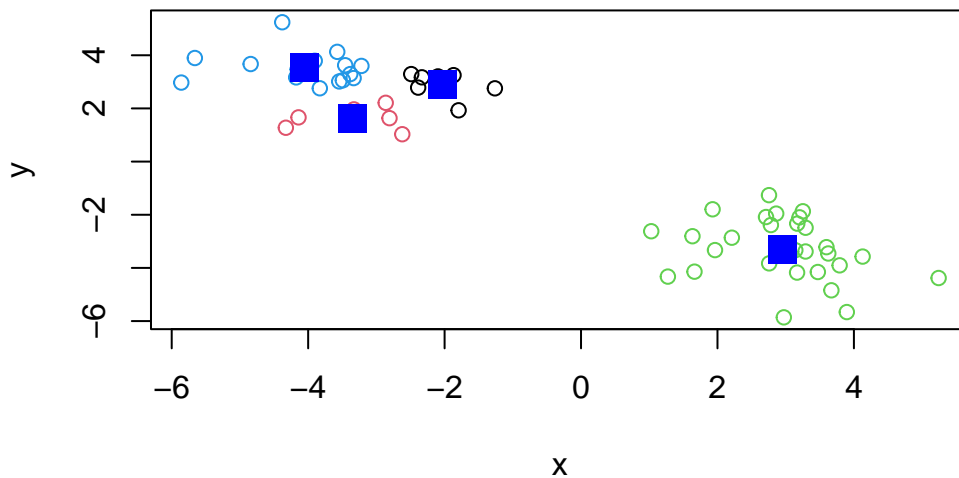
key-point: K-means clustering is super popular but can be misused. One big limitation is that it can impose a clustering pattern on your data even if clear natural group doesn't exist - i.e. it does what you tell it to do in terms of `centers`.

###Hierarchical Clustering

The main function in "base" R for hierarchical clustering is called `hclust()`.

You can't just pass our dataset as it is into `hclust()`. You must give "distance matrix" as input. We can get this from the `dist()` function in R.
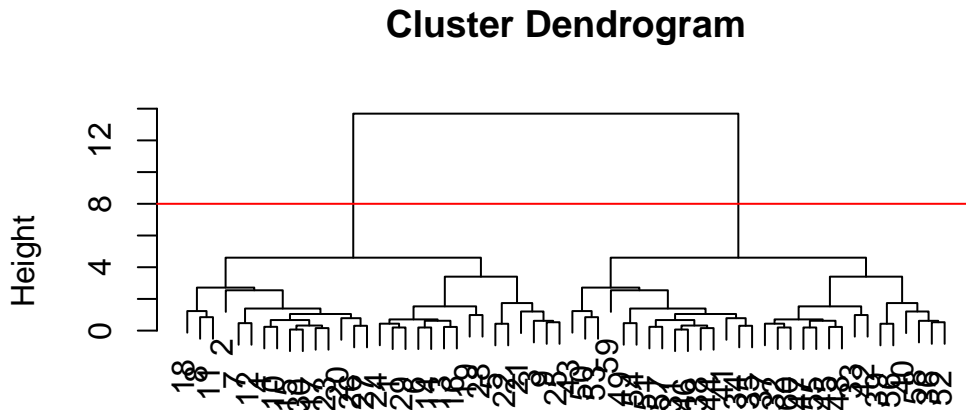
```
d <- dist(x)
hc <- hclust(d)
hc
```

```
Call:
hclust(d = d)

Cluster method   : complete
Distance         : euclidean
Number of objects: 60
```

The results of `hclust()` don't have a useful print method, but do have a special `plot()` method.

```
plot(hc)
abline(h=8,col="red")
```

Cluster Dendrogram

d
hclust (*, "complete")

To get our main cluster assignment (membership vector) we need to "cut" the tree at the big goalposts...
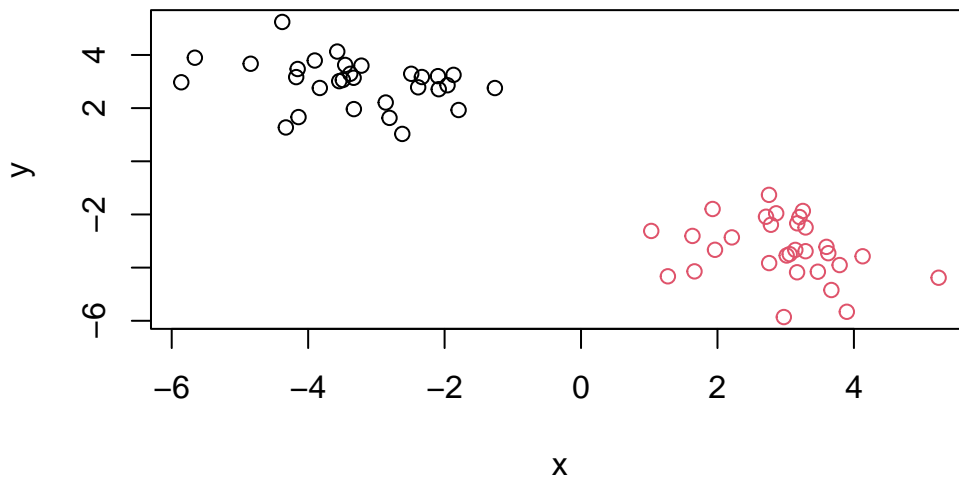
```
grps <- cutree(hc, h=8)
grps
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
table(grps)
```

```
grps
 1  2
30 30
```

```
plot(x, col=grps)
```


Hierarchical clustering is distinct in that the dendrogram (tree figure) can reveal the potential grouping in your data (unlike K-means).

Principal Component Analysis (PCA)

PCA is a common and highly useful dimensionality reduction technique used in many fields - particularly bioinformatics.

Here we will analyze some data from the UK on food consumption.

Data import

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)

head(x)
```

```
          X England Wales Scotland N.Ireland
1        Cheese      105    103      103        66
2  Carcass_meat      245    227      242       267
3    Other_meat      685    803      750       586
4          Fish      147    160      122        93
5 Fats_and_oils      193    235      184       209
6        Sugars      156    175      147       139
```

```
rownames(x) <- x[ ,1]
x <- x[ , -1]
head(x)
```

```
              England Wales Scotland N.Ireland
Cheese            105   103      103        66
Carcass_meat      245   227      242       267
Other_meat        685   803      750       586
Fish              147   160      122        93
Fats_and_oils     193   235      184       209
Sugars            156   175      147       139
```
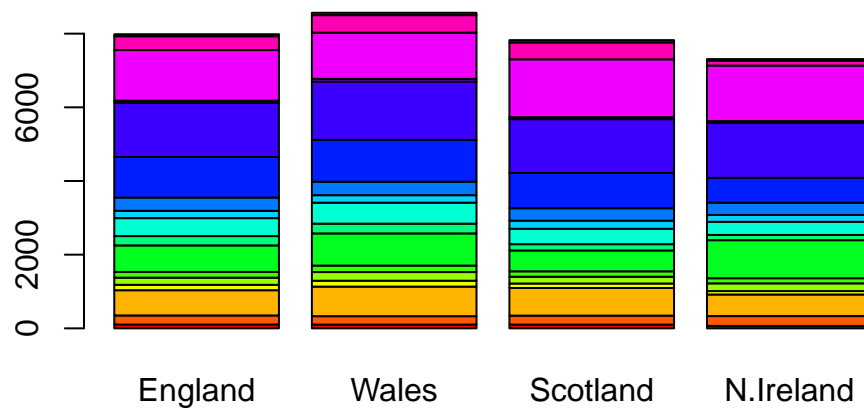
```
x <- read.csv(url, row.names = 1)
head(x)
```

```
              England Wales Scotland N.Ireland
Cheese            105   103      103        66
Carcass_meat      245   227      242       267
Other_meat        685   803      750       586
Fish              147   160      122        93
Fats_and_oils     193   235      184       209
Sugars            156   175      147       139
```

```
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```
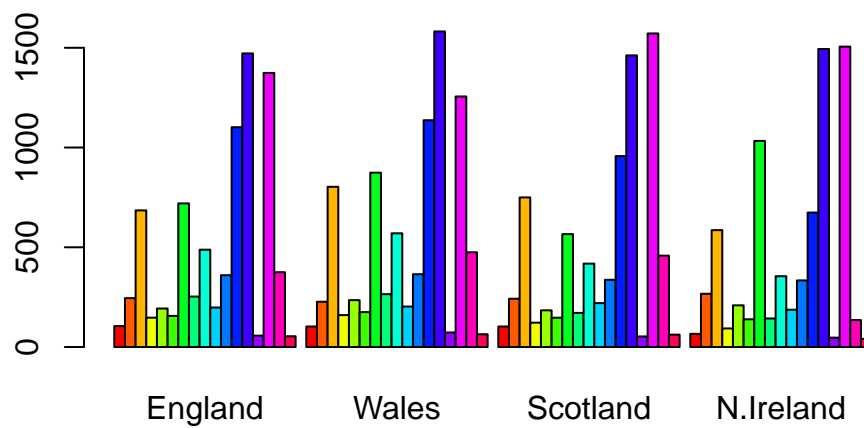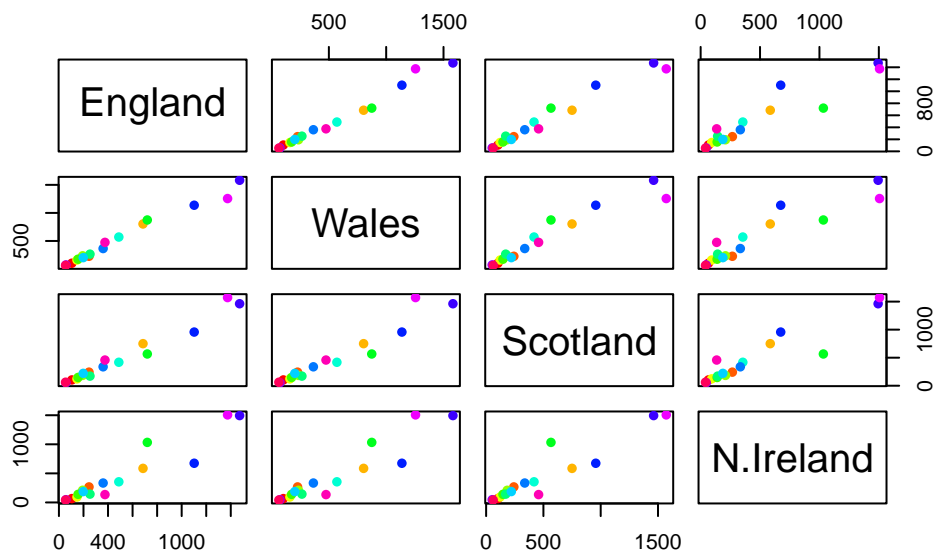
```
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```


One conventional plot that can be useful is called a "paris" plot.

11

```
pairs(x, col=rainbow(nrow(x)), pch=16)
```


PCA to the rescue

The main function in "base" R for PCA prcomp().

```
pca <- prcomp( t(x) )
summary(pca)
```

```
Importance of components:
                          PC1      PC2      PC3       PC4
Standard deviation    324.1502 212.7478 73.87622 2.921e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

The prcomp() function returns a list object of our results with five attributes/components.

```
attributes(pca)
```

```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"


$class
[1] "prcomp"
```

The two main "results" in here are `pca$x` and `pca$rotation`. The first of these (`pca$x`) contains the scores of the data on the new PC axis - we use these to make our "PCA plot".

```
pca$x
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -9.152022e-15
Wales     -240.52915 -224.646925 -56.475555  5.560040e-13
Scotland   -91.86934  286.081786 -44.415495 -6.638419e-13
N.Ireland  477.39164  -58.901862  -4.877895  1.329771e-13
```

```
library(ggplot2)
library(ggrepel)

# Make a plot of pca$x with PC1 vs PC2
ggplot(pca$x) +
  aes(PC1, PC2, label=rownames(pca$x)) +
  geom_point() +
  geom_text_repel()
```
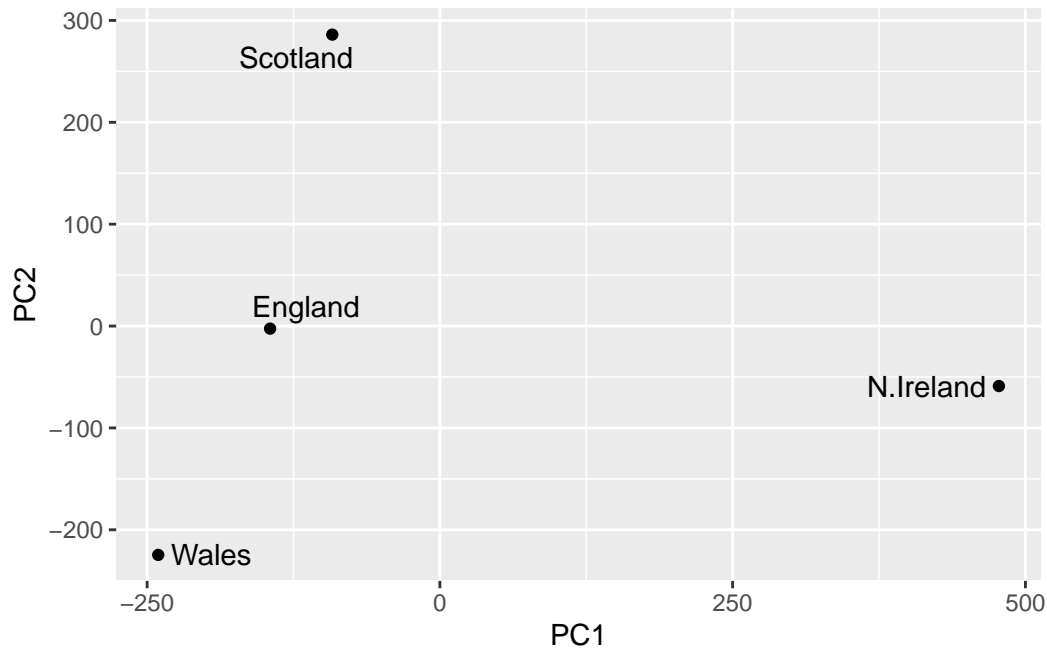
The above plot shows that the eating habits in Ireland are very different compared to England, Wales, and Scotland.

The second major result is contained in the `pca$rotation` object or component. Let's plot this to see what PCA is picking up...

```
pca$rotation
```

```
                           PC1          PC2          PC3          PC4
Cheese             -0.056955380  0.016012850  0.02394295 -0.409382587
Carcass_meat        0.047927628  0.013915823  0.06367111  0.729481922
Other_meat         -0.258916658 -0.015331138 -0.55384854  0.331001134
Fish               -0.084414983 -0.050754947  0.03906481  0.022375878
Fats_and_oils      -0.005193623 -0.095388656 -0.12522257  0.034512161
Sugars             -0.037620983 -0.043021699 -0.03605745  0.024943337
Fresh_potatoes      0.401402060 -0.715017078 -0.20668248  0.021396007
Fresh_Veg          -0.151849942 -0.144900268  0.21382237  0.001606882
Other_Veg          -0.243593729 -0.225450923 -0.05332841  0.031153231
Processed_potatoes -0.026886233  0.042850761 -0.07364902 -0.017379680
Processed_Veg      -0.036488269 -0.045451802  0.05289191  0.021250980
Fresh_fruit        -0.632640898 -0.177740743  0.40012865  0.227657348
Cereals            -0.047702858 -0.212599678 -0.35884921  0.100043319
Beverages          -0.026187756 -0.030560542 -0.04135860 -0.018382072
```

14

```
Soft_drinks          0.232244140   0.555124311 -0.16942648   0.222319484
Alcoholic_drinks    -0.463968168   0.113536523 -0.49858320  -0.273126013
Confectionery       -0.029650201   0.005949921 -0.05232164   0.001890737
```

```
ggplot(pca$rotation) +
  aes(PC1, rownames(pca$rotation))+
  geom_col()
```
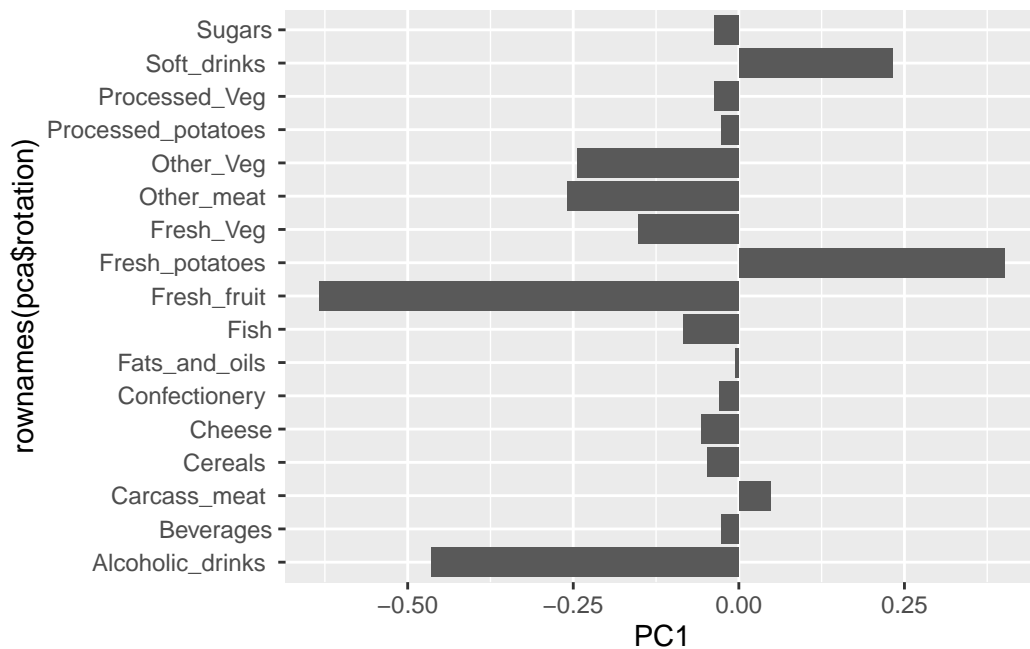

The above plot expands on the previous one. It specifically shows how the eating habits of the people of Ireland are different from England, Wales, and Scotland. For example, Irish people consume more potatoes and soft drinks while the others drink more alcohol and eat more fresh fruit.